

Nombre y apellido: FEDERICO SUMITER

Nº de orden: 6 L.U.: 37/19

1	2	3	Nota
30	35	30	95

(A)

TEORÍA DE LENGUAJES
Segundo cuatrimestre de 2022

Segundo parcial

- El examen dura cuatro horas.
- El examen es a libro abierto. No está permitido utilizar dispositivos electrónicos.
- Se aprueba con 65 puntos sobre 100.
- Resuelva cada ejercicio en hojas separadas.
- Escriba nombre, apellido, L.U. y número de orden en cada hoja. Numere las hojas.
- Consigne por escrito todos los razonamientos que justifiquen sus respuestas.

Ejercicio 1. (30 pts) Sea $G_1 = \langle \{A, B\}, \{a, b\}, P_1, S \rangle$ una gramática extendida, con P_1 :

$$\begin{aligned} S &\rightarrow (aA)^*B \\ A &\rightarrow (A \mid \lambda)a \\ B &\rightarrow b^+B? \end{aligned}$$

- Determinar si G_1 es ELL(1). Si no lo es, dar una gramática ELL(1) que genere el mismo lenguaje que G_1 .
- Dar un párser recursivo-iterativo que reconozca el lenguaje generado por G_1 .

Ejercicio 2. (35 pts) Sea la gramática $G_2 = \langle \{F\}, \{\neg, \vee, \rightarrow, p\}, P_2, F \rangle$, con P_2 :

$$F \rightarrow \neg F \mid F \vee F \mid F \rightarrow F \mid p$$

- Dar su tabla SLR. Señalar los conflictos que posee y de qué tipo son.
- Modificar la tabla para eliminar los conflictos sin alterar el lenguaje reconocido por el párser, teniendo en cuenta que \neg es más precedente que \vee , que asocia a izquierda, y es a su vez más precedente que \rightarrow , que asocia a derecha.

Ejercicio 3. (35 pts) Sea la gramática $G_3 = \langle \{S, T\}, \{(\,, \,), [\,, \,], \text{num}\}, P_3, S \rangle$, con P_3 :

$$\begin{aligned} S &\rightarrow T [\text{num}] \\ T &\rightarrow (T \text{ num } T) \mid \text{num} \end{aligned}$$

Interpretaremos las cadenas del lenguaje generado por G_3 como árboles binarios cuyos nodos contienen números enteros —representados por el token **num**, que posee un atributo *val* provisto por el analizador léxico—, seguidos por un número entero entre corchetes.

Un árbol binario se dice *de búsqueda* si, para todo nodo, el valor del mismo es mayor que todos los valores contenidos en su subárbol izquierdo y menor o igual que todos los valores contenidos en su subárbol derecho.

Dar una gramática de atributos basada en G_3 que

- genere el lenguaje de los árboles binarios que son de búsqueda y contienen como elemento al valor indicado entre corchetes, y además
- sintetice una cadena sobre el alfabeto $\{I, D\}$ que indique el camino a seguir para llegar desde la raíz del árbol al valor indicado entre corchetes, donde I indica bajar hacia la izquierda y D, hacia la derecha.

Por ejemplo: i. la cadena $((1 \ 3 \ 7) \ 8 \ 10) \ [6]$ no pertenece al lenguaje, porque el árbol no contiene el valor 6; ii. la cadena $((1 \ 3 \ 8) \ 7 \ 10) \ [3]$ no pertenece al lenguaje, porque el árbol no es de búsqueda; iii. la cadena $((1 \ 3 \ (4 \ 6 \ 7)) \ 8 \ 10) \ [4]$ sí pertenece al lenguaje, y a partir de ella debe sintetizarse la cadena IDI.

① a).

Para determinar si G_1 es $ELL(1)$, primero voy a desextenderlo para ver si es $LL(1)$

$$S \rightarrow RB$$

$$R \rightarrow aAR | \lambda$$

$$A \rightarrow Ha$$

$$H \rightarrow A | \lambda$$

$$B \rightarrow B_1 B_2$$

$$B_1 \rightarrow bX$$

$$X \rightarrow bX | \lambda$$

$$B_2 \rightarrow B | \lambda$$

RECURSIÓN
A IZQ.
NO IMM.

2 FORMAS DE
GENERAR VARIOS b

UNA VEZ HECHO ESTO ANTECHO LAS PRODUCCIONES PARA VER QUE PROBLEMAS HAY A SIMPLE VISTA. EN PARTICULAR LOS DOS PROBLEMAS MARCADOS OCASIONAN QUE MI GRAMÁTICA NO SEA $LL(1)$, LO QUE IMPLICA QUE NO ES $ELL(1)$.

DICHOS PROBLEMAS⁽²⁾ VISTOS CON MÁS DETALLE SON:

● LA RECURSIÓN A IZQUIERDA NO INMEDIATA

Ejemplo: ~~$S \rightarrow RB \rightarrow SARB \rightarrow \dots$~~
 $A \rightarrow Ha \rightarrow Aa$

● LAS 2 FORMAS DE GENERAR VARIOS b:

ESTO SE PUEDE APRECIAR ANALIZANDO LOS SD DE:

$$SD(X \rightarrow bX) = \{b\}$$

$$SD(X \rightarrow \lambda) = \text{sig}(X) = \text{sig}(B_1) = \text{prim}(B_2)$$

y donde $b \in \text{prim}(B_2)$

∴ Como $\exists \cap \neq \emptyset$ NO ES $LL(1)$

(PARA EL
MISMO NO DEL
CADA IZQ.)

(Y TAMPOCO $ELL(1)$)

➤ VAMOS COMO ANTES UNA GRAMÁTICA QUE GENERE EL MISMO LENGUAJE Y SEA ELL(1). PARA ELLO PRIMERO VOY A AJUSTAR LA GRAMÁTICA Y LUEGO LA VOY A EXTENDER.

LUEGO DE REALIZAR PRUEBAS (EN HOJA BORRADOR) PARA RESOLVER EL EJERCICIO PUEDO NOTAR QUE RESULTA MÁS CÓMODO Y FÁCIL ~~COMPRENDER~~ (EN ESTE CASO) COMPRENDER LA GRAMÁTICA. ✓

SEPARÉMOSLA EN 2 ~~PARTE~~ PARTES:

- POR UN LADO ^{REDE} GENERAR UNA CANTIDAD ∞ DE 'a', ^{SIENDO LA ÚNICA} ~~REDE~~ RESTRICCIÓN QUE GENERA 2 'a' como mínimo $\rightarrow \emptyset \neq \emptyset$
- LUEGO DE ESTO ^{REDE} GENERAR UNA CANTIDAD ∞ DE 'b', SIENDO LA ÚNICA RESTRICCIÓN QUE GENERA 1 'b' como mínimo.

DICHO ESTO, LA GRAMÁTICA QUEDA: (GENERANDO EL MISMO LENGUAJE)

P_1 :

$S \rightarrow aaA b B$

$A \rightarrow aA \mid \lambda$

$B \rightarrow bB \mid \lambda$

$SD(S \rightarrow aaA b B) = \{a\}$

$SD(A \rightarrow aA) = \{a\}$

$SD(A \rightarrow \lambda) = sig(A) = \{b\}$

$SD(B \rightarrow bB) = \{b\}$

$SD(B \rightarrow \lambda) = sig(B) = \{\$ \}$

$G_1 = \langle \{A, B, S\}, \{a, b\}, P_1, S \rangle$ ✓

SU FORMA EXTENDIDA

SERÍA TAL QUE:

P_1 :

$S \rightarrow aaA b B$

$A \rightarrow (aA)?$

$B \rightarrow (bB)?$

Como $\forall (A \rightarrow B, A \rightarrow \gamma)$ con $B \neq \gamma$

$SD(A \rightarrow B) \cap$

$SD(A \rightarrow \gamma) = \emptyset$

puedo decir que es LL(1)

$S \rightarrow aa \overbrace{a^+}^{a^+} \overbrace{b^+}^{b^+} b^*$

Donde es ELL(1) ya que es LL(1).

(SIGUE EJERCICIO 1)

b).

PARTE DEL PROCESO S SIN NECESIDAD DE UTILIZAR UN
S' YA QUE EL MISMO NO ES LLAMADO ~~RECURSIVO~~ EN LAS
↓ PRODUCCIONES

(• UN "MAIN")

VOY A HACER EL PROCESO RECURSIVO ITERATIVO DE:

 $S \rightarrow a a A b B$ $A \rightarrow (aA)?$ $B \rightarrow (bB)?$

• Proc S():

match('a');

match('a');

A();

match('b');

B();

match('\$');

accept();

End

• Proc A():

if (tc == 'a') {

match('a');

A();

}

End

• Proc B():

if (tc == 'b') {

match('b');

B();

}

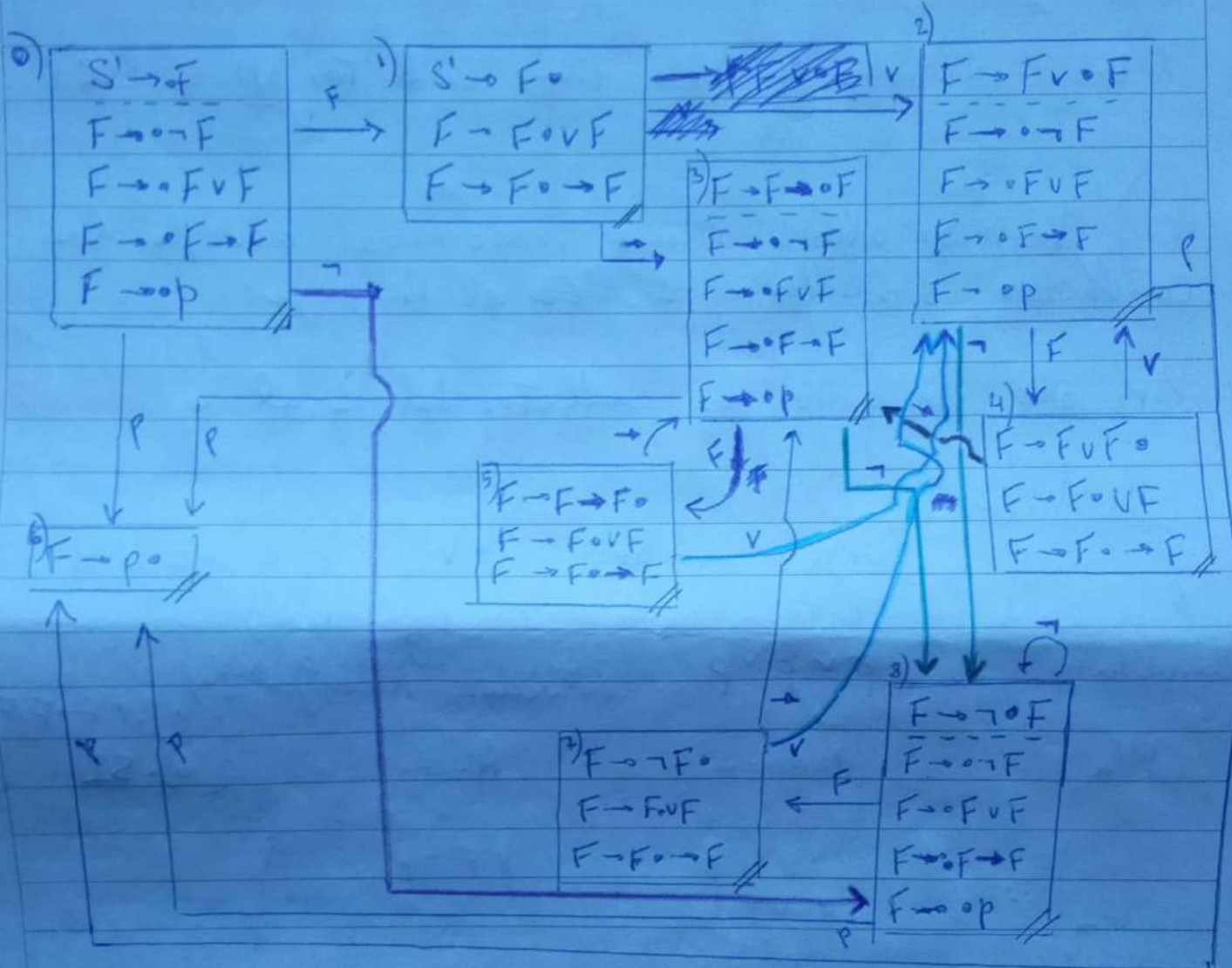
End

2 a).

¡Muy bien! :)

(Algunos tienen colores para distinguirlas)

PARA ARMAR LA TABLA CLR PRIMERO ARMAMOS EL AUTÓMATA:



Estado	\neg	v	\rightarrow	P	$\$$	F
0	S8			S6		1
1		S2	S3		ACCEPT	
2	S8			S6		4
3	S8			S6		5
4		1) $v: F \rightarrow FvF$ S2	2) $v: F \rightarrow FvF$ S3		$v: F \rightarrow FvF$	
5		3) $v: F \rightarrow Fp$ S2	4) $v: F \rightarrow Fp$ S3		$v: F \rightarrow Fp$	
6		$v: F \rightarrow p$	$v: F \rightarrow p$		$v: F \rightarrow p$	
7		5) $v: F \rightarrow \neg F$ S2	6) $v: F \rightarrow \neg F$ S3		$v: F \rightarrow \neg F$	
8	S8			S6		7

Calculo: $\text{sig}(F) = \{v, \rightarrow, \$\}$

Hay 6 conflictos en total. Todos del tipo shift-reduce

Aclaración: Como la tabla tiene conflictos No es SLR.

b). Veamos ahora como podemos arreglar los conflictos sin alterar ~~la tabla~~ el lenguaje:

Para ello, enumeramos los conflictos del 1 al 6 **EN NARANJA**

Conflicto 1

Surge cuando: v en el estado 4

Analizando (con la técnica de mirar para atrás) una cadena candidata es $pvpvp$ (TODAS LAS CADENAS DE ABAJO EN ADELANTE LAS CARGO ASI)

Pila	Cadena	Acción
0	$pvpvp\$$	S6
Op6	$vpvp\$$	$v:F \rightarrow p$
OF1	$vpvp\$$	S2
OF1v2	$pvp\$$	S6
OF1v2p6	$vp\$$	$v:F \rightarrow p$
OF1v2F4	$vp\$$	Conflicto 1

a) $v:F \rightarrow FvF$
b) S2

a) OF1v2F4	$vp\$$	$v:F \rightarrow FvF$
OF1	$vp\$$	S2
OF1v2	$p\$$	S6
OF1v2p6	$\$$	$v:F \rightarrow p$
OF1v2F4	$\$$	$v:F \rightarrow FvF$
OF1	$\$$	ACCEPT

b) OF1 v 2 F4	vp\$	S2
OF1 v 2 F4 v2	p\$	S6
OF1 v 2 F4 v2 p6	\$	v: F → p
OF1 v 2 F4 v2 F4	\$	v: F → FvF
OF1 v 2 F4	\$	v: F → FvF
OF1	\$	ACCEPT.

Con a) tenemos que asocia a i39.

Con b) tenemos que asocia a der.

Por enunciado me quedo con

1.A

Conflicto 2

Surge cuando \rightarrow en el estado 4

Cadena candidata: $p \vee p \rightarrow p$

Pila	Cadena	acción
0	$p \vee p \rightarrow p$	S6
Op6	$v \rightarrow p$	v: F → p
OF1	$v \rightarrow p$	S2
OF1 v 2	$p \rightarrow p$	S6
OF1 v 2 p6	$\rightarrow p$	v: F → p
OF1 v 2 F4	$\rightarrow p$	Conflicto 2

a). v: F → FvF

b). S3

Dejo de separar y analizar en cosas por falta de tiempo. Me acón en adelante.

Acá el conflicto está en reducir FvF o seguir shiftando.
Como v es más presente que \rightarrow
tomo 2.A

Conflicto 3 Surge cuando $\bullet v$ en el estado S
Cadena candidata: $p \rightarrow p \vee p \checkmark$

Pila	Cadena	Acción
0	$p \rightarrow p \vee p \$$	S6
Op6	$\rightarrow p \vee p \$$	$r: F \rightarrow p$
DF1	$\rightarrow p \vee p \$$	S3
OF1 $\rightarrow 3$	$p \vee p \$$	S6
OF1 $\rightarrow 3p6$	$\vee p \$$	$r: F \rightarrow p$
OF1 $\rightarrow 3F5$	$\vee p \$$	Conflicto 3 a). $F \rightarrow F \rightarrow F$ b). S2.

Acá el conflicto es parecido al anterior. Como queremos que el \vee sea más precedente tomamos (3.B) para poder shiftear y lograr esto. ~~para poder shiftear~~

Conflicto 4 Surge cuando $\vee \rightarrow$ en el estado S
Cadena candidata: $p \rightarrow p \rightarrow p \checkmark$

Pila	Cadena	Acción
0	$p \rightarrow p \rightarrow p \$$	S6
Op6	$\rightarrow p \rightarrow p \$$	$r: F \rightarrow p$
DF1	$\rightarrow p \rightarrow p \$$	S3
OF1 $\rightarrow 3$	$p \rightarrow p \$$	S6
OF1 $\rightarrow 3p6$	$\rightarrow p \$$	$r: F \rightarrow p$
OF1 $\rightarrow 3F5$	$\rightarrow p \$$	Conflicto 4 a). $F \rightarrow F \rightarrow F$ b). S3

Acá el conflicto recae en cómo asociar:

$(p \rightarrow p) \rightarrow p$ o $p \rightarrow (p \rightarrow p)$. Como queremos que asocie a derecha tomamos (4.B)

LU: 37/19

MAIA: 5 DE 6

N.º DE SEQU: 6

FEDERICO
SUAITER

Conflicto 5: Surge cuando $\bullet V$ en el estado 7
Cadena candidata: $\neg p \vee p$ ✓

Pila	Cadena	Acción
\emptyset	$\neg p \vee p \$$	S8
$0 \rightarrow 8$	$p \vee p \$$	S6
$0 \rightarrow 8 p 6$	$\vee p \$$	$\vee: F \rightarrow p$
$0 \rightarrow 8 F 7$	$\vee p \$$	Conflicto 5 $\begin{cases} a). \vee: F \rightarrow \neg F \\ b). S2 \end{cases}$

Acá el conflicto surge de no saber si consumir $\neg F$ para volver a F $(\vee: F \rightarrow \neg F)$ o seguir para luego reducir $F \vee F$ $(\vee: F \rightarrow F)$. Por enunciado sabemos que \neg tiene mayor precedencia, \therefore tomar (5.A) ✓

Conflicto 6: Surge cuando $\bullet \rightarrow$ en el estado 7
Cadena candidata: $\neg p \rightarrow p$

Pila	Cadena	Acción
\emptyset	$\neg p \rightarrow p \$$	S8
$0 \rightarrow 8$	$p \rightarrow p \$$	S6
$0 \rightarrow 8 p 6$	$\rightarrow p \$$	$\vee: F \rightarrow p$
$0 \rightarrow 8 F 7$	$\rightarrow p \$$	Conflicto 6 $\begin{cases} a). \vee: F \rightarrow \neg F \\ b). S3 \end{cases}$

Acá el conflicto surge de forma muy parecida al anterior. Por enunciado sabemos que \neg tiene mayor precedencia, \therefore tomar (6.A) ✓

DE ESTA FORMA, TOMANDO 1.A, 2.A, 3.B, 4.B, 5.A, 6.A (QUITANDO LAS OPCIONES NO SELECCIONADAS) LOGRAMOS ELIMINAR LOS CONFLICTOS SIN ALTERAR EL LENGUAJE ✓

Asamblea

(EXPLICACIÓN DETRÁS)

S \rightarrow T [num] $\left\{ \begin{array}{l} T.x = \text{num.val}; \\ S.\text{path} = T.\text{path}; \\ \text{Condition } (T.\text{found} == \text{true}); \end{array} \right.$

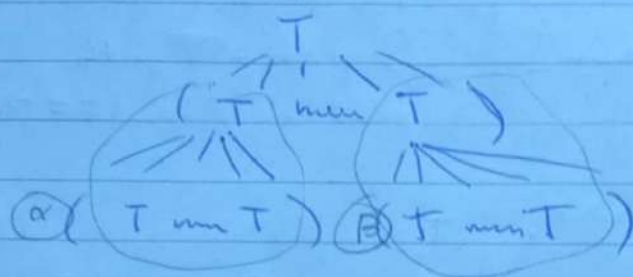
T \rightarrow (T₁ num T₂) $\left\{ \begin{array}{l} T.\text{path} = \left\{ \begin{array}{l} T_1.x = T.x \text{ } T_2.x = T.x; \\ T.\text{found} = T_1.\text{found} \vee T_2.\text{found}; \\ \text{if } (T_1.\text{found}) \text{ then } 'I'++ T_1.\text{path} \\ \text{else if } (T_2.\text{found}) \text{ then } 'D'++ T_2.\text{path} \\ \text{else } " " \text{ endif}; \end{array} \right. \\ \text{Condition } (T_1.\text{max} < \text{num.val} \wedge T_2.\text{min} > \text{num.val}) \\ T.\text{max} = T_2.\text{max}; \\ T.\text{min} = T_1.\text{min}; \end{array} \right.$

T \rightarrow num $\left\{ \begin{array}{l} T.\text{found} = \text{if } (\text{num.val} == T.x) \text{ then TRUE else FALSE} \\ T.\text{path} = " "; \\ T.\text{min} = \text{num.val}; \\ T.\text{max} = \text{num.val}; \end{array} \right.$

Símbolo	Atributo	Tipo de dato	Tipo de Atrib.
num	val	int	SINTETIZADO
T	found	bool	SINTETIZADO
T	x	int	HEREDADO
T	min	int	SINT.
T	max	int	SINT.
T	path	string	SINT.
S	path	string	SINT.

UTILICÉ T.x PARA PODER "BAJAR" EL VALOR ENTRE CORCHETES EN MI ÁRBOL. DE ESTA FORMA VOY A PODER OBTENER 'SI LO ENCUENTRO' MEDIANTE T.found QUE VA A IR SUBIENDO PARA LUEGO DETERMINAR SI EFECTIVAMENTE EL VALOR ESTÁ EN MI ÁRBOL. POR OTRO LADO, UTILICÉ T.min y T.max PARA PODER

COMPARAR DE FORMA ADECUADA LOS VALORES EN EL NODO DE MI ÁRBOL. ESTO LO HAGO ASÍ PUESTO QUE SI TENEMOS:



En α y β A PRIORI NO SE SI ESTOY A IZQ. O DERECHA, ENTONCES ME GUSTO AMBOS VALORES EN T. ~~PERO~~

Por lo PEDIDO EN LA DEF. DE ÁRBOL DE BÚSQUEDA, COMPARO QUE EL VALOR MÁS GRANDE DE T_1 SEA MENOR QUE $\min.val$ Y QUE EL VALOR MÁS CHICO DE T_2 SEA MAYOR QUE $\min.val$. LUEGO SI ESTA CONDICIÓN SE CUMPLE, TOMO COMO EL MÁS GRANDE DE T $T_2.max$ Y EL MÁS CHICO DE T $T_1.min$. (ESTO ÚLTIMO VALE X EL CHEQUEO DE LA COND. ADemás DE SER CONDICIÓN NECESARIA)

HASTA AHORA CUMPO CON LA DEF. PEDIDA DE ÁRBOL DE BÚSQUEDA, PERO FALTA LA CHENA. PARA ELLO, UNA VEZ ENCONTRADO EL VALOR, APROVECHO EL T.found ANTES MENCIONADO PARA IR REMANDO POCO A POCO EL T.path. EN EL A MEDIDA QUE VOY SUBIENDO, VOY AGREGANDO ADELANTE ~~DE~~ LA DIRECCIÓN QUE TOQUE, PARA QUE LUEGO A LA HORA DE QUERER RECORRER EL CAMINO OBTENIDO EN S.path PUEDA IR SIGUIENDO LOS PASOS DE IZQUIERDA A DERECHA.

TAL COMO VIMOS EN CLASE (DEFORM. PARECIDA) SI QUITAMOS LA PARTE DE $[min]$ PODAMOS VER QUE NUESTRA GRAMÁTICA ES L-ATRIBUIDA. USANDO ESTA IDEA DE "OBTENER EL VALOR DE LA IZQ." UNA SOLA VEZ EN LA RAÍZ DEL ÁRBOL PODAMOS DE CERTA FORMA GARANTIZAR QUE \exists UN ORDEN TOROLÓGICO.